Enabling Comprehensive Data-Driven System Management for Large Computational Facilities

James C. Browne ‡, Robert L. DeLeon*, Charng-Da Lu*, Matthew D. Jones*, Steven M. Gallo*, Amin Ghadersohi*, Abani K. Patra*, William L. Barth†, John Hammond†, Thomas R. Furlani*, Robert T. McLay†

*Center for Computational Research, SUNY at Buffalo, Buffalo, NY
clu2@fastmail.us, browne@cs.utexas.edu

‡Department of Computer Science, University of Texas, Austin, TX
{rldeleon, furlani, smgallo, jonesm, ag28, abani}@buffalo.edu

†Texas Advanced Computing Center, University of Texas, Austin, TX
john.hammond@intel.com, bbarth@tacc.utexas.edu

ABSTRACT

This paper presents a tool chain, based on the open source tool TACC_Stats, for systematic and comprehensive job level resource use measurement for large cluster computers, and its incorporation into XDMoD, a reporting and analytics framework for resource management that targets meeting the information needs of users, application developers, systems administrators, systems management and funding managers. Accounting, scheduler and event logs are integrated with system performance data from TACC_Stats. TACC_Stats periodically records resource use including many hardware counters for each job running on each node. Furthermore, system level metrics are obtained through aggregation of the node (job) level data. Analysis of this data generates many types of standard and custom reports and even a limited predictive capability that has not previously been available for open-source, Linux-based software systems. This paper presents case studies of information that can be applied for effective resource management. We believe this system to be the first fully comprehensive system for supporting the information needs of all stakeholders in open-source software based HPC systems.

1. INTRODUCTION

The high cost (capital, power and manpower) of HPC resources requires us to optimize the usage – an outcome that is only possible if adequate data is collected and is used to drive systems management with good reporting and analysis/prediction at different granularities – job, application, user and system. While limited capabilities of this type [1] have been available in proprietary systems, there exists no complete solution for open systems. *This paper presents a tool chain for systematic and comprehensive job level resource use measurement, and its incorporation into XDMoD, a reporting and analytics framework that targets meeting the information needs of users, application developers, systems administrators, systems management and funding managers. Availability of such reporting will allow the identification of poor resource usage and guide improvements.*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

SC13 November 17-21, 2013, Denver, CO, USA
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2378-9/13/11...$15.00.
http://dx.doi.org/10.1145/2503210.2503230

1.1 Motivation

A comprehensive system which addresses the information needs of users, application developers, systems administrators, resource managers, and program officers will enable all of the stakeholders to more efficiently and effectively utilize existing systems, plan for new systems, determine the overall efficiency with which a given resource is being used and the effectiveness with which it is meeting programmatic goals, and assess the return on investment. For example, with such a system, users will be able to see how their use of different types of system resources compares to the system-wide averages, either for the system as a whole or for users working in the same or similar fields of engineering and science. Users will also be able to determine which systems their jobs will execute on with maximum efficiency. Resource managers can identify if the applications are well tuned to the architecture and are making efficient use of the resource. Systems administrators will have support tools for diagnosing system faults and failures and for assessing the effectiveness with which the current scheduling and resource management policies and tactics are obtaining desired objectives and suggestions for possible improvements. System planners will have a comprehensive basis for assessing trends in resource use and in designing systems to meet specific or general goals. HPC systems are purchased based on performance on a projected job mix, which may in fact be significantly different from what is actually experienced. Thus operational efficiencies are often far lower than projections.

Identification of problematic resource usage and guiding improvements will need a data driven framework. The XDMoD (**X**SE**D**E **M**etrics **o**n **D**emand) tool is being developed to provide such a comprehensive resource management framework for XSEDE and high-performance computing centers [2]. To date the focus of XDMoD has been primarily on the development of usage, system performance, and scientific impact metrics. However, the ability to provide detailed information on the performance (CPU and memory usage, swapping/paging activities, network bandwidth, filesystem usage, and interconnect fabric traffic) of all applications running on these systems is necessary for effective systems management. To achieve this capability, the SUPReMM (Integrated HPC **S**ystems **U**sage and **P**erformance of **Re**sources **M**onitoring and **M**odeling) program integrates the TACC_Stats tool data into XDMoD. TACC_Stats periodically samples node and system state values and a wide spectrum of hardware counters to provide a rich (and extensive) dataset of performance data for every application run on each node in a given HPC cluster for analysis and reporting.

1.2 Challenges

The tools for measurement and analysis of resource use that are available in the standard Linux environment are not comprehensive and do not address the information needs of many of the stakeholders described above. For example, the standard systat/SAR measurement system does not resolve resource use by job or by user. While total system vendors (eg. Cray, IBM and HP) use Linux based software stacks, they often enhance the default measurement and analysis tools with proprietary extensions that are not usable elsewhere. Another significant challenge is the sheer volume of the data that must be addressed when we deal with system information at the granularity of jobs sampled frequently. The use of standard Linux logs and measurement tools for data also comes with special challenges since they are gathered and reported in many different formats [3].

1.3 Approach

Our approach is systematic integration of analysis tools and data acquisition. The measurements we utilize are combined data from event logs, schedulers, performance counters etc. gathered by several open source and customizable new tools. These new tools address major deficiencies in the measurement capabilities provided in the open source Linux software stack. The new tools include, TACC_Stats, the replacement of systat/SAR, which resolves resource use by processor core, by job, and by user. They also include a rationalized version of syslog that adds job ID information to each message and also maps all of the diverse message types generated by the software stack into a single uniform format. Another tool called Lariat, generates unified summary data on the execution of a job such as which libraries are called.

The function of the SUPReMM program is to integrate these data sources into XDMoD, thereby creating a comprehensive resource management framework for Linux-based HPC systems. *In fact, as far as we have found, this is the first attempt to systematically incorporate system and usage data to meet monitoring and management requirements of all stakeholders of large computer systems.* Further the XDMoD/SUPReMM system is extensible to different system architectures.

The rest of the paper includes a detailed description of the SUPReMM tools and reporting and analysis of generated data from two XSEDE resources Lonestar4 and Ranger. Section 2 discusses related work and Section 3 gives a brief overview of how TACC_Stats works. Section 4 discusses the information needs of each stakeholder classes and gives examples of analyses and reports and the value generated for each stakeholder class. Finally, Section 5 gives conclusions and future work.

2. RELATED WORK

Del Vento *et al*. [1] adopted a similar approach to tackling inefficient HPC resource utilization. DeVento [1] primarily targeted optimization of user codes and applications. Supremm also generates reports on job and application resource use which identify jobs and applications codes which have anomalous levels of use of one or more resources, indicating a need for optimization. Supremm, however, has the much broader target of comprehensive support for all stakeholders in a large system. Also, the measurements of DeVento [1] are based on proprietary systems (IBM POWER/AIX) while SUPREMM is designed for open-source software based HPC clusters.

Del Vento et al. [1] report that they were able, with user cooperation, to improve CPU usage (e.g. correct process affinity/CPU binding) and troubleshoot thorny issues (e.g. memory leak). Once a job or application with a pattern of inefficient use of one or more resources has been identified by the SUPREMM analyses and reports, we recommend that the user or application developer (with collaboration from consulting staff) apply one of the many performance optimization tools available for open source clusters: Tau[4,5,6], HPCToolkit[7,8], IPM[9], Open or SpeedShop[10], Scalasca[11]. VTune[12], Paradyn [13] and PerfExpert[14,15].

There are, in addition to the Linux built-in monitors such as systat/SAR and ionstat, many open source and commercial monitoring resource management tools. The open source tools include: CLUMON [1], PCP [2], Ganglia [3], Nagios [4], NEWT [5], Lorenz [6], and SLURM [7]. Splunk [8] is a commonly used commercial system. Each of these tools tools has one or more serious deficiencies with respect to comprehensive job/core level resource management. None of these tools, with the exception of CLUMON, which gets its data from PCP, resolve the data by job at the core level which is critical for comprehensive resource management that includes researchers, application developers and policy makers. CLUMON/PCP and SLURM do not monitor hardware performance counters. Ganglia and Nagios do not resolve by job or user.

TACC_Stats, however, collects data from performance counters (per core and socket), block device statistics (per device), scheduler accounting (per CPU), InfiniBand usage, Lustre filesystem usage (per mount), Lustre network usage, memory usage (per socket), network device usage (per device), NUMA statistics (per socket), process statistics (ps), SysV shared memory segment usage, ram-backed filesystem usage (per mount), dentry/file/inode cache usage virtual memory statistics.

In summary, while much of the data collected by TACC_Stats can be obtained by combining data from multiple sources and/or using the capabilities provided in some tools for adding measurement monitors, such an approach using multiple monitors with multiple data formats would greatly add to the overhead and complexity of measurement. Having a single job-oriented comprehensive source of data simplifies both collection and data analysis.

3. RESOURCE MEASUREMENT

The Linux systat package is a comprehensive collection of performance monitoring utilities, each of which reports resource statistics of specific components of a system in its own format. TACC_Stats enhances sysstat/sar for the open source software based HPC environment in many ways. It is a single executable binary that covers all performance measurement functions of sysstat and outputs in a unified, consistent, and self-describing plain-text format. It is batch job aware: Performance data are tagged with a batch job id to enable offline job-by-job profile analysis. It supports newer Linux counters and hardware devices. Its source code [24] is also highly modular and can be easily extended to gather new kinds of performance metrics.

Currently TACC_Stats can gather core-level CPU usage (user time, system time, idle, *etc.*), socket-level memory usage (free, used, cached, *etc.*), swapping/paging activities, system load and process statistics, network and block device counters, interprocess communications (SysV IPC), software/hardware interrupt request (IRQ) count, filesystems usage (NFS, Lustre, Panasas), interconnect fabric traffic (InfiniBand and Myrinet), and CPU hardware performance counters. For a complete list of the data acquired by TACC_Stats, see the TACC_Stats web site [25].

TACC_Stats executes at the beginning of a job, periodically during the job (currently every ten minutes) and at the end of the job. At this frequency of execution, TACC_Stats generates an overhead of approximately 0.1%. Before beginning each job, TACC_Stats reprograms the performance counters it uses. On AMD Opteron, the events are FLOPS, memory accesses, data cache fills and SMP/NUMA traffic. On Intel Nehalem/Westmere, the events are FLOPS, SMP/NUMA traffic, and L1 data cache hits. At the periodic invocations, TACC_Stats only reads values from performance registers without reprogramming them to avoid overriding measurements initiated by users while ignoring user set counters. The work-flow chart, Figure 1, shows a schematic of how the collected data is analyzed organized, input into XDMoD and reported.

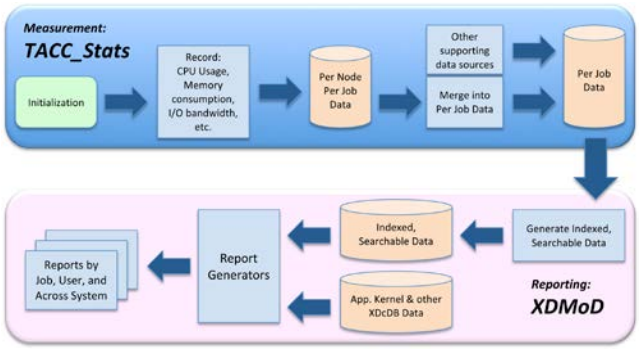


Figure 1. Work flow chart detailing integration of the TACC_Stats and other data sources, such as Lariat, into the XDMoD Framework for resource management.

4. ANALYSES AND REPORTS

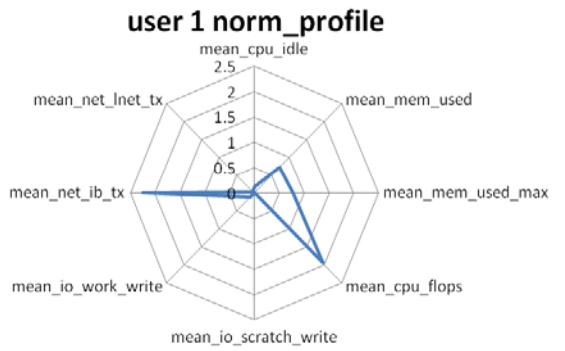
This section describes how SUPReMM is applied, defines the metrics used in the example reports and illustrates the capabilities of the system and its potential use by all stakeholders.

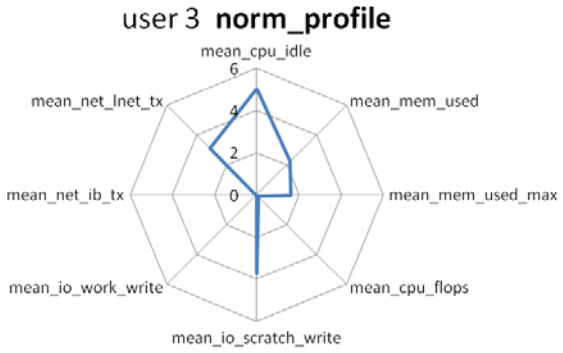
4.1 Data Sources

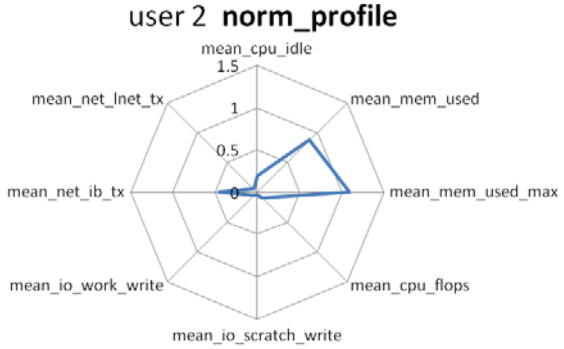
The case studies given as examples in this paper were carried out on the Ranger and Lonestar4 supercomputers at the Texas Advanced Computing Center (TACC). Ranger (decommissioned as of February 2013) is a Linux cluster with 3936 compute nodes, each of which has four 2.3GHz AMD Opteron quad-core processors (16 cores in total) and 32 GB of memory. The filesystem is Lustre and the interconnect is InfiniBand. Lonestar4 is also a Linux cluster with 1088 Dell PowerEdgeM610 compute nodes. Each compute node has two Intel Xeon 5680 series 3.33GHz hexa-core processors and 24 GB of memory. Lonestar4 has two filesystems: Lustre and NFS and its interconnect is InfiniBand (NFS is connected via Ethernet).

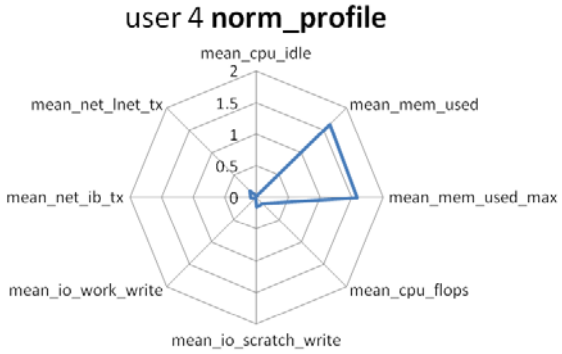
TACC_Stats has been deployed on Ranger for 20 months and on Lonestar4 since November of 2011. On Ranger, TACC_Stats generates a raw data file of 0.5 MB per node per day and collectively 60 GB (uncompressed) or 20 GB (compressed) for the entire cluster per month.

We analyzed TACC_Stats data collected on Ranger during June 2011 to January 2013 with a total of 521,010 jobs, and on Lonestar4 from November 2011 to January 2013 with a total of 337,011 jobs. The jobs included in this study are those longer than the default TACC_Stats sampling interval of 10 minutes. The values were calculated by the job weighted by node*hour. The job pool for our analysis contains predominantly XSEDE jobs with a small proportion of non-XSEDE jobs (e.g. from TACC partners and Texas higher educational institutions). We ingested both the raw TACC_Stats output files and job accounting information into an IBM Netezza data warehouse appliance and a MySQL database, respectively.









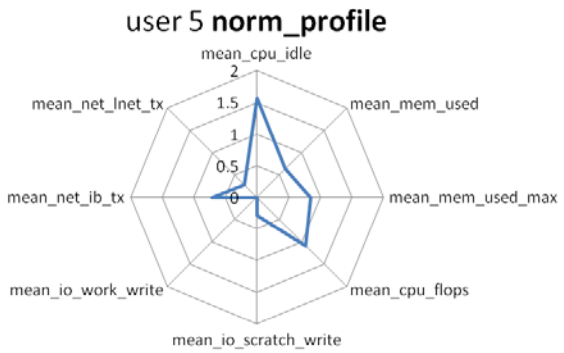


Figure 2. Sample user usage profiles based on 8 TACC_Stats metrics for 5 heavy users of Ranger. The plots show normalized by the average value of each metric over all of the **Value** usage on Ranger i.e. average user = 1. Values above 1 indicate heavier resource usage; values below 1 indicate lighter usage. Note the variability in the usage profiles between users.

4.2 Overview of Example Analyses and Reports

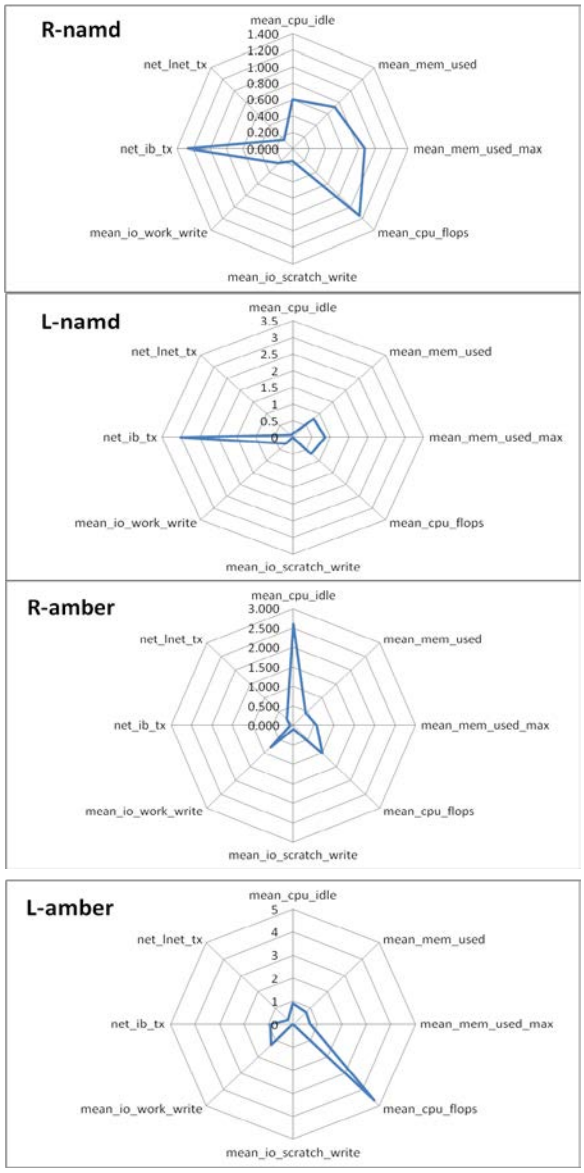
The analyses and reports used as examples will be mainly concerned with eight key metrics and closely related quantities: cpu_idle, mem_used, mem_used_max, cpu_flops, io_scratch_write, io_work_write, net_ib_tx, and net_lnet_tx. We have chosen these eight based on a correlation analysis over *all* of the measured metrics. We found that there are many highly correlated or anti-correlated metrics, such as cpu_user is negatively correlated to cpu_idle, or net_ib_rx is positively correlated to net_ib_tx. Therefore, we have selected the smallest independent set of metrics that describe the execution behavior of the job mix on each system. The definitions of the eight metrics are as follows. Cpu_idle is that fraction of the CPU not utilized by the job in user space or by the system. Mem_used refers to the per node memory used, including the disk buffer and cache managed by the Linux operating system Mem_used_max refers to the peak observed memory usage over all used nodes and all sampling intervals of a job. Cpu_flops is the floating-point operations per second produced on a job. Io_scratch_write and io_work_write refer to writing from the scratch and work file systems (Lustre). The difference of these arises in the purge policy and quota size. "Scratch" is purged periodically and has a largish quota to the tune of hundreds of TB, and "work" is non-purged space with a 200GB quota. Net_ib_tx, and net_lnet_tx refer to network data transmission rates.

4.3 Analyses and Reports by Stakeholder

The XDMOD system has a powerful and flexible analysis interface that has many analyses reports preprogrammed and also the option for stakeholders to define custom reports. The reports listed here for each stakeholder, are only a small subset of the analyses and reports that are possible.

4.3.1 *USER* Analyses and Reports: Sample reports of interest are – resource use profile by job, comparative resource use by user, jobs with anomalous or inefficient resource use patterns, Bottleneck identification and performance optimization and job completion failure profiles.

Proposition: Users are often unaware of the resource use characteristics of their jobs – in particular inefficient usage. These reports provide a simple way for users to determine their relative efficiency on different systems and enable them to choose the system best suited for their application. Anomalous resource use patterns may be an indicator of undetected bugs in a program. They are also commonly the precursors of job failures from exception conditions and may sometimes induce system hangups though soft lockups of compute nodes which eventually lead to job-wide node-level hangups. Importantly, inefficient resource use patterns by a job, such as high cpu idle fraction, call attention to potential opportunities to improve the performance of the application, and in so doing benefit the user who obtains greater throughput, and the user community by freeing up busy resources.



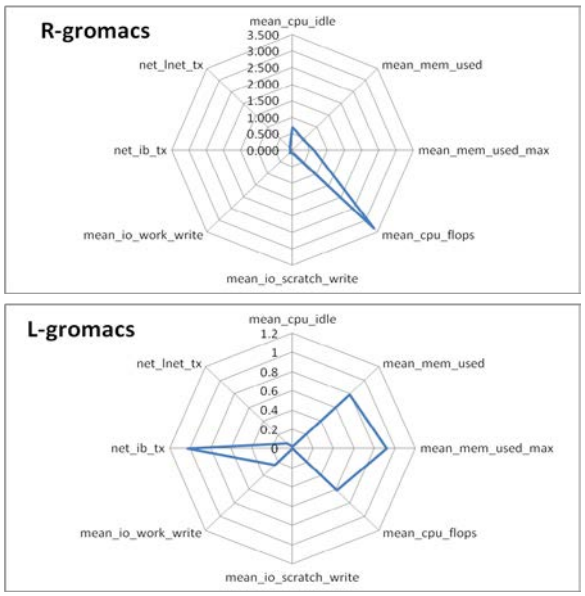


Figure 3. Resource usage profiles for eight TACC_Stats metrics for the three molecular dynamics (MD) applications, NAMD, AMBER, and GROMACS on Ranger (R-) and Lonestar4 (L-). Plots have been normalized by the average value of each metric calculated over all jobs (i.e. average =1) on Ranger and Lonestar4, respectively. Values above 1.0 indicate heavier resource usage; values below 1.0 indicate lighter usage.

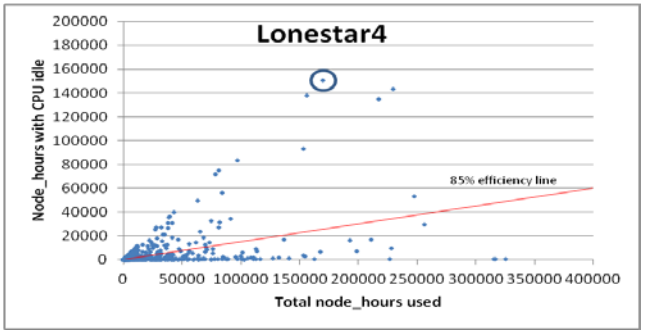
***User usage profiling*:** The TACC_Stats data can be used to generate a profile of usage for each job, user, application or allocation/program/charge number. Over the period studied, ~2000 users submitted jobs to Ranger. We have characterized this usage in terms of the 8 key metrics previously described. Other metrics could be added or substituted as desired. Figure 2 shows a comparison of the usage profile of 5 heavy users of Ranger. The profiles have been normalized by dividing by the average values for the particular metric calculated over all users. Therefore, a typical user would have a value of one for each of the 8 metrics and this would appear as a perfect octagon with each vertex at unity in the radar chart. Values above one indicate heavy usage: below one, light usage of the particular resource or performance metric. There is a great variation in the usage profile of these 5 users in spite of the fact that they were selected as the largest consumers of node-hours. User 1 has a high level of use of FLOPS and high network traffic use. User 3 has a very high Cpu idle fraction and a high fraction of Lustre file system traffic, indicating jobs dominated by by IO.

4.3.2 *APPLICATION DEVELOPER* Report(s): In addition to the user reports, developers will have reporting available on anomalous or inefficient resource use patterns, basic resource use profiles, comparative resource use profiles, and variability of resource use.

Value Proposition: A steadily increasing fraction of scientific and engineering computation is being done with "community" application codes – a development likely to increase as agencies support more of these tools. Developers who know how efficiently their application executes on different platforms and how effectively (with respect to resource use) users apply the application can improve applications and user training. These reports enable the application developers to have both detailed knowledge at the level of each execution by each user and also cumulative comparative data on execution behaviors on each system and across systems. The reports can also enable the developers to identify the circumstances where the application has terminated abnormally. Figure 3 compares the resource use profiles of the three most used molecular dynamics codes on Ranger and Lonestar4, NAMD, AMBER, and GROMACS. In the plots in Figure 3 we compare the performance of NAMD, AMBER, and GROMACS, relative to the performance of the average job run on Ranger and Lonestar4. Note that the plots have different scales. If the metric of cpu idle fraction is used to determine efficiency, then NAMD and GROMACS run more efficiently than AMBER on both Ranger and Lonestar4. The NAMD usage pattern on Ranger and Lonestar4 is very similar whereas GROMACS and AMBER usage is different on the two clusters. These plots suggest examining AMBER to investigate the variation in the floating point and cpu idle metrics.

4.3.3 *SUPPORT STAFF* Report(s): Primary reports of interest here are – jobs or user with anomalous or inefficient resource use patterns, jobs with abnormal termination condition, and major application resource use profiles.

Value Proposition: These reports will enable facility-based support staff to identify jobs and applications which can benefit from attention or are in imminent danger of failure. They also provide information that can be used to assist in diagnosing and correcting job execution problems referred to them by users – a function that will often need the same reports that support staff will use. In addition to giving support staff the information they need to respond to user queries, the reports enable a transition to a proactive role. If a major user has a resource use profile that has an anomalous value for a given metric, for example cpu_idle, then the computing center support staff may wish to contact that user to see if they can assist them in achieving better performance for their applications. Figure 4 shows user resource use patterns for all jobs. Figure 4 is based on the usage from all users on Ranger or Lonestar4 for their respective study periods. It shows "wasted" node-hours, that is, those spent with an idle CPU, vs total node-hours consumed. While in some cases the user could choose to run on only a fraction of the cores on each node in order to take advantage of more memory per core used, in fact very few of these cases were found and most idle node-hours are indeed wasted. On the Ranger plot, the red line shows the 90% efficiency mark, that is, below this line less than 10% of the CPU hours consumed are spent in idle. The 90% mark was chosen because this is the approximate average value over all Ranger users. For the Lonestar4 plot the 85% efficiency line is shown to reflect the average CPU idle % over all Lonestar4 usage. While many large CPU-hour users have very efficient codes, as evident by the large number of users down near the bottom of the chart below the 90% or 85% efficiency line, many large users are spending 50% or more of their node-hours in CPU idle mode. Using this information, service providers could contact the users with poorly performing applications to determine whether their efficiency can be improved. For example, for the Ranger and Lonestar4 plots in Figure 4, the users circled have spent 87% and 89% respectively of the consumed node hours in CPU idle. Figure 5 shows their usage profiles. Note that other than the large CPU user idle which is 8 and 5 times larger than that for the average Ranger or Lonestar4 user, there is only normal to light usage of other resources such as memory, file systems and network, that is, nothing to explain the anomalously high CPU idle fraction. There are clearly outliers that may benefit from closer scrutiny.



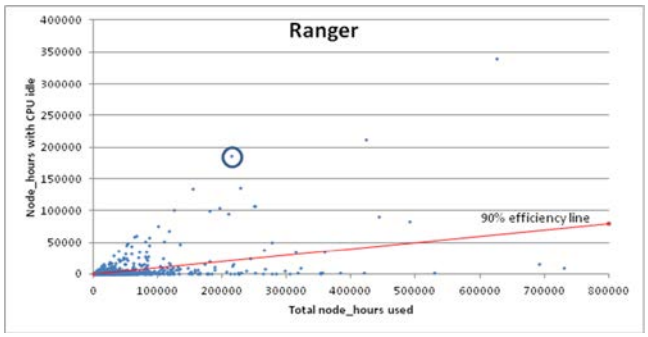


Figure 4. Plot of node-hour usage vs wasted node-hours. Wasted node-hours are defined to be those spent with the CPU idle. The average efficiency of jobs on Ranger are 90% and Lonestar4 85%, where we define efficiency to be the percentage of time not spent in CPU idle. We have marked the average efficiency lines on the Ranger and Lonestar4 plots with a red line. On each plot a single problematic user has been circled.

4.3.4 *SYSTEMS ADMINISTRATORS* Reports(s): Administrators are likely to be interested in diagnostic reports for job/system faults/failures, system level resource use patterns and workload characterization, job/application characteristics, job completion failure profiles, and, fault/failure diagnostic reports.

Value Proposition: Currently, most of the tasks of systems administration are accomplished using custom site specific scripts written independently at each installation with little sharing of approaches or reuse for solving common problems which include diagnosing system faults and failures, determining "optimal" settings for system software such as job schedulers and evaluating the efficiency and effectiveness of new versions of the system software stack. These reports will simplify and ease each of these tasks. The ANCOR tool [26]**,** which is not discussed in this paper, combines TACC_Stats data with rationalized logs [27] to generate analyses and reports which diagnose the possible causes of system faults and failures. Reports based on job resource use patterns and workload characterization can assist in determining the effectiveness of software stack updates and optimizing job scheduling.

The illustration used for potential value to systems administration is the persistence characteristics of resource use patterns over time i.e. how well can one predict future performance/usage on Ranger? Starting from a given point in time can we predict what will occur 10 minutes, 100 minutes, 1000 minutes, or 10,000 minutes into the future? To a certain extent we can because there is a well-defined persistence of the performance metrics.

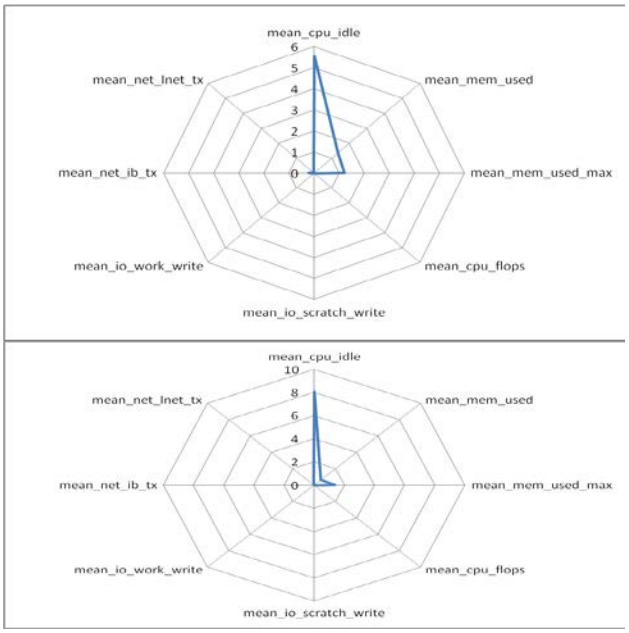


Figure 5. User usage profile for the users circled in Figure 4. The first chart is for Ranger, the second for Lonestar4. These cases were selected for the high CPU idle fraction. Other metrics indicate normal resource usage. For these cases there is no obvious other resource usage to explain the high idle fraction.

The method that we used was to compute and examine the standard deviation of a given metric. We can introduce an offset, for example X minutes, and take the difference between the offset values and the original values and look at the standard deviation of this difference. This tells us how similar that metric is X minutes later. If there is no tendency to persist, the standard deviation should be approximately equal to the original standard deviation of the metric. If the value at X minutes is nearly identical to the original value, the standard deviation will approach 0, and we have a very good idea what value the metric will be. As the offset is increased to very large values, eventually the standard deviation of the difference will approach the original standard deviation of the metric and we cannot predict the value any better than using the general statistics of the ensemble.

We have applied this method for 5 metrics: cpu_flops, mem_used, io_scratch_write, net_ib_tx and cpu_idle. Table 1 shows the results. Each table entry is the offset standard deviation divided by the original standard deviation. Obviously the ability to predict the next value of the metric 10 minutes later is very good with the standard deviation decreasing almost by a factor of ten. At 100 minutes there is still a good memory of the original value but by 1000 minutes there is little memory of the original value. There are small differences in the persistence between the 5 metrics. The ability to predict the future value of the metric goes in increasing order: io_scratch_write < net_ib_tx ~ cpu_idle < mem_used ~ cpu_flops. However, overall they are all well fit by a logarithmic model. Aside from the reversal of net_ib_tx and io_scratch_write, the predictive ability order follows the descending value of the coefficient of variation of the metrics. A

logarithmic model can be used to fit the data. The last row in Table 1 shows the R^2 values of the fit to each metric.

Offset(min)	flops	mem	write	ib_tx	cpu_idle
10	0.123	0.148	0.311	0.268	0.267
30	0.211	0.217	0.494	0.431	0.375
100	0.377	0.344	0.670	0.652	0.544
500	0.705	0.638	0.999	0.911	0.849
1000	0.889	0.814		0.999	1.009
Fit R^2	0.98	0.95	0.995	0.998	0.98

Table 1. Persistence data for 5 metrics

All 5 metrics can be combined on a single fit when the standard deviations at the offset times are normalized by dividing by the standard deviation of the original metric as was done in Table 1. The upper plot in Figure 6 shows this fit. The intercept is -0.17(6) p=0.016, the slope is 0.36(2) p=5 E-12, R^2=0.87. While it is obviously not quite as good as the original fits, indicating a real difference between the metrics, it still serves to show some similarity in the predictability of the various metrics. A similar analysis can be done for Lonestar4, see the lower plot in Figure 6. The values for the Lonestar4 fit are very similar to the Ranger fit; the intercept is -0.28(5) p=2E-5, the slope is 0.42(2) p=9 E-15, R^2=0.93. One would expect the persistence to be related to the average job length. The longer the jobs the longer we would expect the persistence to be and the easier it should be to predict the value of the metrics further into the future. For Ranger the average weighted job length over this period was 549 minutes. The model agrees qualitatively with this value in that below 549 minutes we can predict the future value of the metric with some accuracy, above this value there is relatively little predictive ability. For Lonestar4 the average job length is slightly shorter 446 minutes, in agreement with the fact that the slope of the Lonestar4 fit is slightly greater.

Ultimately, modeling usage persistence could be a viable strategy to manage resource usage across an HPC cluster. If the usage profile of various applications or users is established, the present usage could be assessed and jobs could be selected from the queue to complement the present resource usage e.g. add high I/O jobs when I/O is relatively free.

4.3.5: ***RESOURCE MANAGERS*** **Reports**: Workload characterization; Job resource use characteristics; Job-level resource use trends; Major application use and resource use characteristics; System-wide resource use profiles; Major application use profiles; Resource use trends and predictions

Value Proposition: System management tasks such as reporting to funding agencies, evaluating the effectiveness of resource operation, and planning for future systems are directly supported by the routine availability of the reports named above. Almost any type of job or application level data (such as job profiles, application use, discipline area use, differences in job characteristics by discipline area, system efficiency measured by use of different types of resources, etc.) are available. Support for planning is provided by workload type trends workload resource use trends, characterization of workloads in terms of use of different resource use patterns, etc.

For illustrative purposes and to provide an indication of the wide range of job and application level data available to resource managers, we present here a broad set of systems level reports on resource use. Figures 7 through 12 are a sample of the many analyses of resource use at the system level. These system level reports are of the most use to system managers and funding agencies reviewing the machine performance and machine productivity. The discussions following each figure sketch the information content of these reports. These reports, and similar reports showing trends in resource use over time, can be important not only in evaluating current performance but for planning enhancements to existing systems and planning for new systems.

Figure 7 shows three sample reports produced by using the TACC_stats data collected from the Ranger machine. In the remainder of this section we also will use the TACC_Stats data to analyze performance.

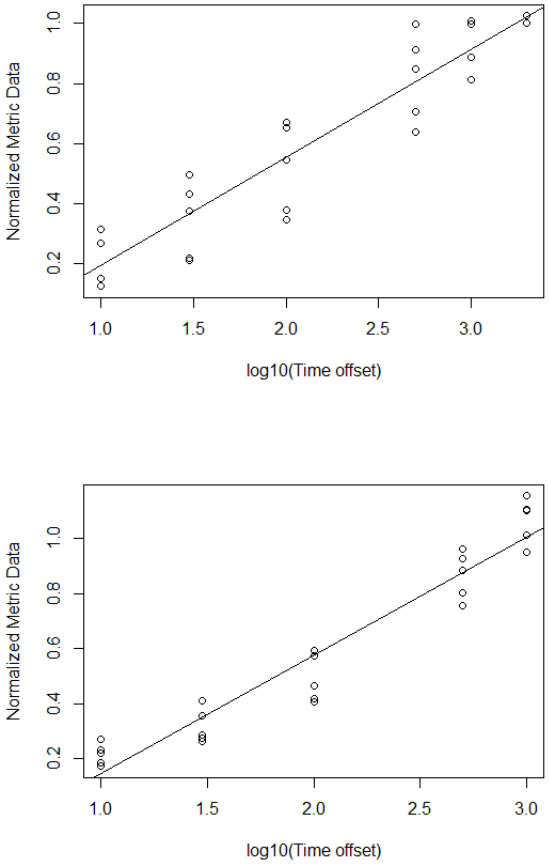


Figure 6. Persistence model fit of all 5 metrics. The standard deviation of each metric normalized by the non-offset standard deviation has been fit against the time offset. The upper chart is for Ranger, the lower chart is for Lonestar4. The plots serve to show how long the value of metrics tends to persist with time.

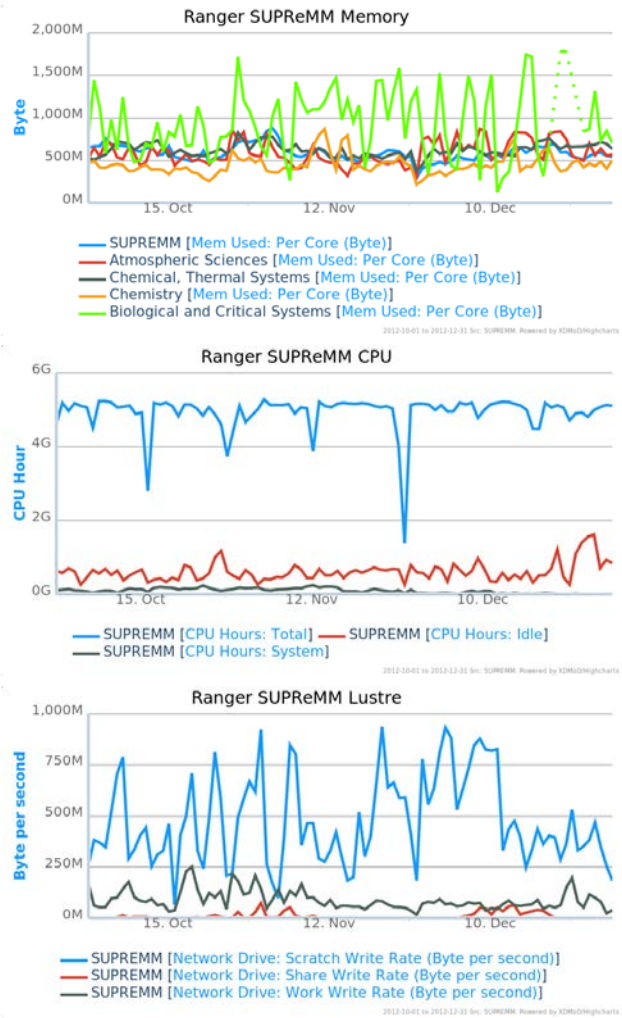


Figure 7. Sample reports based on the data collected by TACC_Stats, processed and published through XDMOD. a) Average memory usage per core for the Ranger machine (XSEDE) and broken up by parent science, b) CPU hours for Ranger (user, idle and system) c) Lustre file system performance on Ranger for scratch, share and work drives.

The variation in the number of active nodes over time is useful to measure as it shows the up/down time of the CPU cluster and can provide a general indication of the efficiency of the scheduler. Although it is not unique to the TACC_Stats analysis, in Figure 8 we present the number of active nodes as a function of time throughout the study period June 2011 to January 2013 on Ranger and November 2011 to January 2013 for Lonestar4. Ranger has 3936 nodes and Lonesatar4 has 1888 nodes, most of which were active throughout the study period with occasional scheduled and un-scheduled shut down periods.

The long-term real-world flops performance of Ranger as measured by TACC_Stats in SSE flops is shown in Figure 9. Ranger was benchmarked to deliver a peak performance of 579 TF. Examination of Figure 9 shows that the actual performance was less than 20 TF, which in of itself is not a particularly surprising result as it is well know real-world performance is much less than peak performance for most applications. Even peak values were less than 50 TF. Lonestar4 flops measured were also quite variable and not comparable to the Ranger plot because they were not SSE flops.

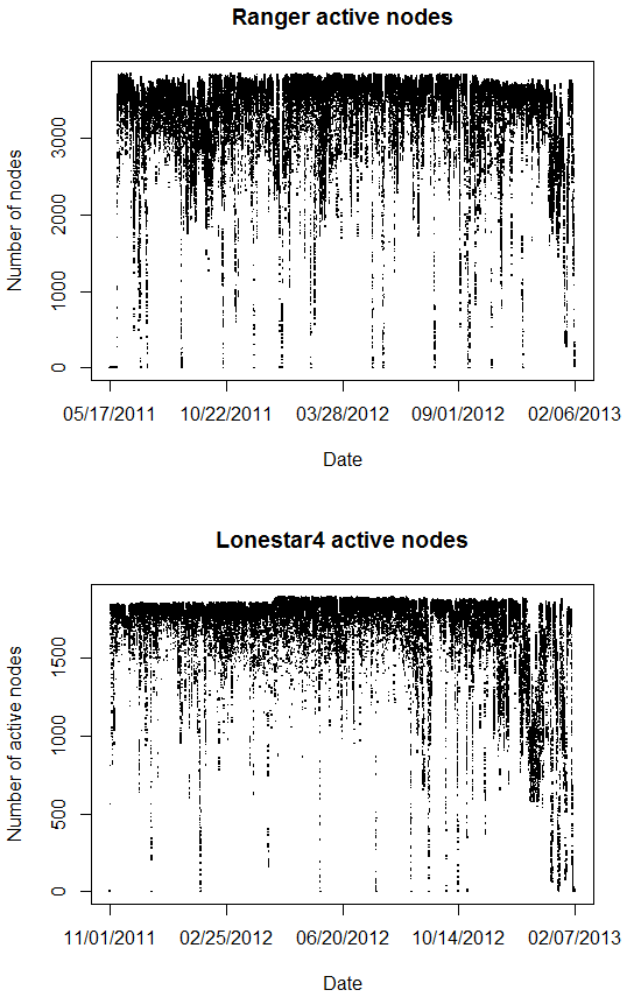


Figure 8: Number of nodes active as a function of time throughout the analysis period June 2011 to January 2013 for Ranger and November 2011 to January 2013 for Lonestar4. The number of active nodes drops to zero during planned or unplanned shutdowns that are relatively infrequent. The smaller variations occur as nodes finish jobs and await new assignment.

Figure 10 shows the distribution of the flops data. Figure 10 (and also Figure 12) show the kernel density [28] (produced by the R statistical software environment) rather than a histogram in order to avoid making binning choices. Typically less than 20 TF were generated by the actual job mix run with very moderate peak values. The small peak at zero is due to shutdown periods. It should be noted that these values are integrated over the various job span and detected by TACC_Stats at 10-minute intervals. The actual instantaneous peak values could be a bit greater.

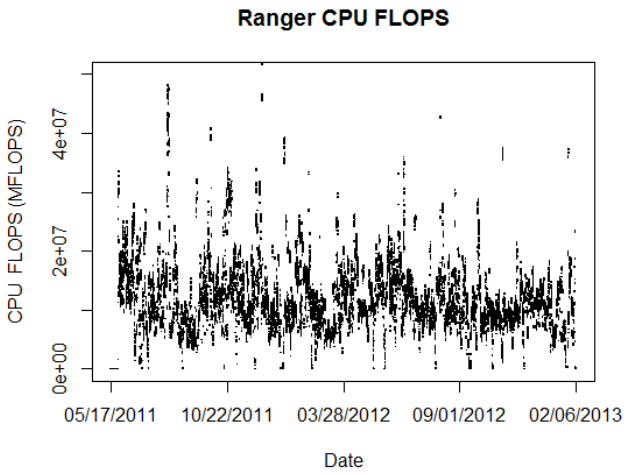


Figure 9. Number of SSE FLOPS produced by Ranger as a function of time, throughout the analysis period June 2011 to January 2013. Note that the output is quite irregular and only a small fraction of the benchmarked peak performance.

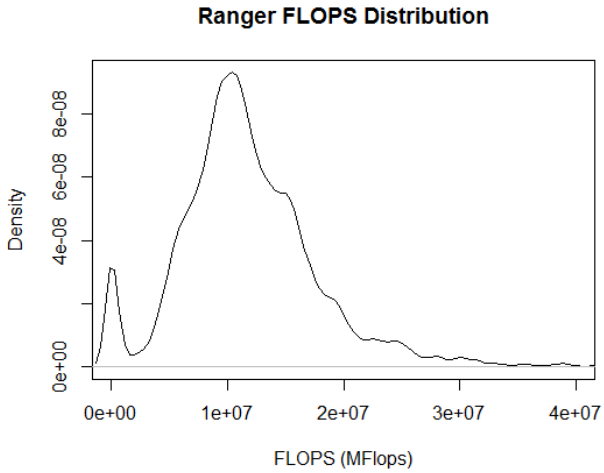
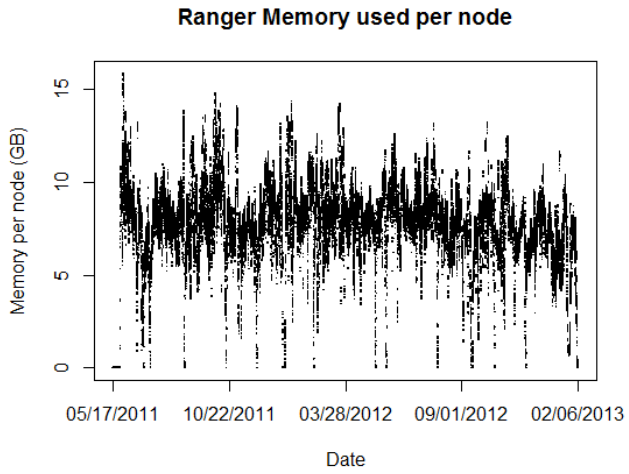


Figure 10. FLOPS distribution from the data shown in Figure 9 for Ranger. Over the course of the study period ranger averaged less than 20 TF, compared to its benchmarked peak performance value of 579 TF.

Ranger has 32 GB of memory for the 16 cores per node. The actual memory usage per node as a function of time is shown in Figure 11. Note that typically less than 10 GB per node is in use at any given time. Even peak values are less than 16 GB per node, that is, less than 50% of the available memory. Figure 12 shows the distribution of the memory usage. It is easy to see the peak is less than 10 GB with negligible usage above 16 GB. The red curve is based on the maximum usage during a job. Even this value is only about 50% of the memory capacity. The same caution that the instantaneous values might be greater made for flops also applies to memory usage. Lonestar4 has 24 GB of memory for the 12 cores per node. Lonestar4 memory usage was higher than that of Ranger. Although average memory used per job per node was still only a bit above 50% of capacity (14/24 GB), the maximum memory used during a job does approach the actual system capacity.



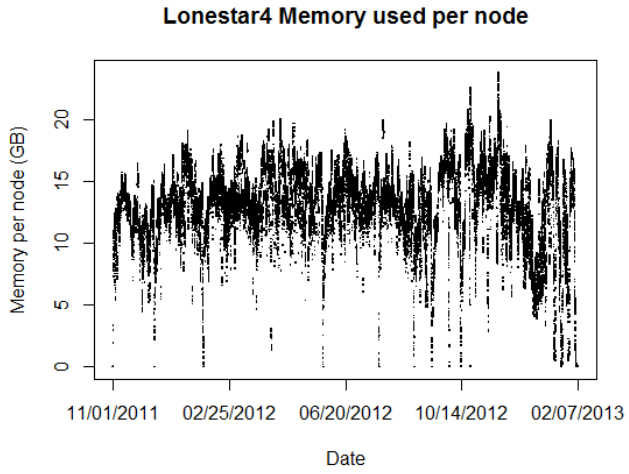


Figure 11: Memory used per node as a function of time throughout the analysis period. Ranger has 32 GB per node. The average value is less than 10 GB, and even peak values are less than 16 GB. Lonestar4 has 24 GB per node and the usage is relatively higher, ~15 GB on average peaking up to 20 GB.

4.3.6 ***FUNDING AGENCIES*** **Reports**: Reports of interest here are system operation profiles; system resource use profiles; discipline area application workload characterization; resource use trends by application area.

Value Proposition: The information needed by funding agencies is similar to the information requirements of systems managers but range across all of the systems for which a funding agency is responsible. Reports illustrating value delivered to funding agencies include: (i) resource use by discipline domains and by widely used application codes, (ii) fractions of resources which are effectively applied by system, (iii) patterns of resource use by discipline and trends in resource use by applications and at the system level. Examples of these reports, Figures 6 through 9, have been presented and discussed in the preceding section. These reports will meet the needs of funding agencies for accountability reporting, for determining the effectiveness of

different systems and approaches to system management and in planning for future procurements.

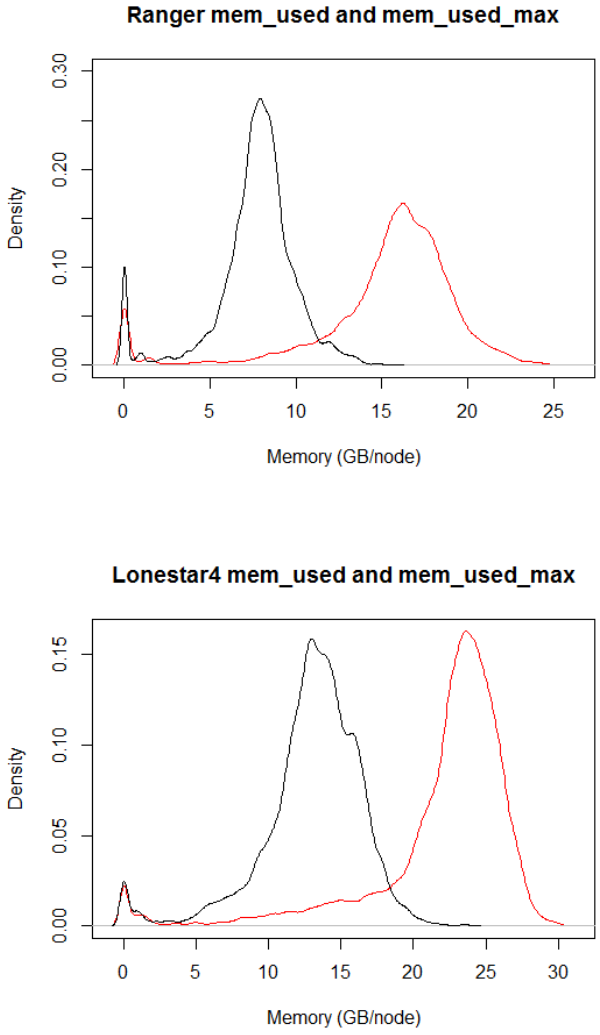


Figure 12. Distribution of memory used per node on Ranger and Lonestar4. The black line is based on the average usage during a job as shown in the data displayed in Figure 11, the red line is based on the maximum usage during a job. Note that for Ranger the memory usage is low, less than 50% of capacity even for the maximum memory usage during the jobs. For Lonestar4 the usage is significantly higher. On average 50% is of the memory is used and nearly the full capacity is used at the job maximum.

5. CONCLUSIONS & FUTURE WORK

The primary focus of this paper has been to demonstrate the ability of combining the new data available from TACC_Stats with standard system data sources such as job scheduler logs to provide detailed job, user, application and system level information and enable analytics of the workload running on a given HPC resource. While characterizing resource utilization at the job level and system levels is useful in its own right, the real power of comprehensive data based resource management lies in the ability to tune system and application performance based on this data to improve overall resource utilization. Given the over-request of most if not all HPC resources, this capability is particularly desirable. Accordingly, future work will center on a detailed investigation of some of the application and system inefficiencies uncovered by the studies carried out here.

For example, we are contacting the owners of particularly poorly running jobs to better understand their workflow and determine if steps can be taken to mitigate the underperformance. These steps are likely to include a wide range of remedies, some simple to implement and others requiring a more substantial effort. For example, the analysis has been done of key TACC_Stats metrics for many commonly run applications on Lonestar4 and Ranger, for one case see Figure 3. This analysis indicates that the NAMD molecular dynamics (MD) package is more efficient than other MD packages such as AMBER. HPC centers might choose to encourage users to consider NAMD for their simulations. Furthermore, although it is hardly surprising to learn that some applications run considerably better on certain machine architectures, with the present tools we can easily identify those applications and provide incentives for users to run on architectures best suited for their application. Additionally, the analysis shown in Figures 2, 4 and 5 of can be used to determine which users are using the resources efficiently and which are not. Such quantitative information can be used to improve their usage and input for a more appropriate allocation for such users.

The data, especially the comparative analysis across architectures and usage classes, raise interesting questions. For example one could argue that given the very different demands placed on machines by different applications and from users from different fields of science, XSEDE should consider providing a “bouquet” of machines tuned to different user groups rather than the monolithic general purpose machines of today.

Although we only briefly alluded to the challenges of analyzing the data generated by a TACC_Stats, we are assessing various technologies (e.g. NoSQL, MONGOdb) to quickly process, store, and query massive TACC_Stats data. This is critical, as it is a key step to developing a capability to rapidly import TACC_Stats data into XDMoD, which will greatly expand its access to end users, systems administrators, and center directors. As a first step, we are presently incorporating the Lonestar4 and Ranger data into XDMoD; the next release at XSEDE13 will feature a TACC_Stats data realm and will therefore be widely available.

TACC_Stats will soon be deployed on TACC’s Stampede and ultimately on all XSEDE resources. Finally, in order to have a broad impact on the efficient operation of HPC resources throughout the U.S., including academic and industrial HPC centers, plans are underway to release an open source version of XDMoD by the end of 2013 that will ultimately have the full resource management capabilities described in this paper.

6. ACKNOWLEGMENTS

This research is supported by the National Science Foundation under award numbers 1203560, 0959870, 1203604, and OCI 1025159.

REFERENCES

[1] Del Vento, D., Engel, T., Ghosh S., Hart, D., Kelly, R., Liu, S. and Valent, R. "System-level monitoring of floating-point performance to improve effective system utilization." In

2011 International Conference for High Performance Computing, Networking, Storage and Analysis (*SC11*)

[2] Furlani, T.R., Jones, M.D., Gallo, S.M., Bruno, A.E., Lu, C.-D., Ghadersohi, A., Gentner, R.J., Patra, A., DeLeon, R.L., von Laszewski, G., Wang, F., and Zimmerman, A., "Performance metrics and auditing framework using application kernels for high performance computer systems," Concurrency and Computation: Practice and Experience, 2012. [Online] Available: http://dx.doi.org/10.1002/cpe.2871

XDMoD: http://xdmod.ccr.buffalo.edu

[3] http://www.thegeekstuff.com/2011/12/linux-performance-monitoring-tools/

[4] K. A. Huck, A. D. Malony, S. Shende, and A. Morris. "Knowledge Support and Automation for Performance Analysis with PerfExplorer 2.0." Large-Scale Programming Tools and Environments, Special Issue of Scientific Programming, vol. 16, no. 2-3, pp. 123-134. 2008.

[5] S. Shende and A. Malony. "The Tau Parallel Performance System." International Journal of High Performance Computing Applications, 20(2): 287-311.

[6] Tau: http://www.cs.uoregon.edu/research/tau/home.php.

[7] N. R. Tallent, J. M. Mellor-Crummey, L. Adhianto, M.W. Fagan, and M. Krentel. "HPCToolkit: performance tools for scientific computing." Journal of Physics: Conference Series, 125. 2008.

[8] L. Djoudi, D. Barthou, P. Carribault, C. Lemuet, J.-T. Acquaviva, and W. Jalby. "Exploring Application Performance: a New Tool for a Static/Dynamic Approach." The Sixth Los Alamos Computer Science Institute Symp. 2005.

[9] HPCToolkit: http://www.hpctoolkit.org/. Last accessed April 1, 2011.

[10] Open|SpeedShop: http://www.openspeedshop.org/wp/.

[11] M. Geimer, P. Saviankou, A. Strube, Z. Szebenyi, F. Wolf, B. J. N. Wylie: Further improving the scalability of the Scalasca toolset. In Proc. of PARA 2010: State of the Art in Scientific and Parallel Computing, Part II: Minisymposium Scalable tools for High Performance Computing, Reykjavik, Iceland, June 6–9 2010, volume 7134 of Lecture Notes in Computer Science, pages 463–474, Springer, 2012.

[12] VTune: software.intel.com/intel-vtune-amplifier-xe

[13] B. P. Miller, M. D. Callaghan, J. M. Cargille, J. K. Hollingsworth, R. B. Irvin, K. L. Karavanic, K. Kunchithapadam, and T. Newhall. "The Paradyn Parallel Performance Measurement Tool." IEEE Computer, 28:37-46. 1995.

[14] M. Burtscher, B.D. Kim, J. Diamond, J. McCalpin, L. Koesterke, and J. Browne. "PerfExpert: An Easy-to-Use Performance Diagnosis Tool for HPC Applications." SC 2010 Int. Conference for High-Performance Computing, Networking, Storage and Analysis. November 2010.

[15] PerfExpert: http://www.tacc.utexas.edu/perfexpert/. .

[16] http://clumon.ncsa.illinois.edu/

[17] http://oss.sgi.com/projects/pcp/

[18] http://ganglia.sourceforge.net/

[19] http://www.nagios.org/

[20] http://www.komodolabs.com/

[21] https://e-reports-ext.llnl.gov/pdf/754636.pdf

[22] https://computing.llnl.gov/linux/slurm/overview.html

[23] http://www.splunk.com/

[24] http://github.com/TACCProjects/tacc_stats

[25] Hammond, J. "TACC_stats: I/O performance monitoring for the intransigent" In *2011 Workshop for Interfaces and Architectures for Scientific Data Storage (IASDS 2011)*

[26] Edward Chuah_, Arshad Jhumka_, Sai Narasimhamurthy, John Hammond, James C. Browne, Bill Barth Linking Resource Usage Anomalies with System Failures from Cluster Log Data (Submitted to SRDS 2013 – Preprint available from mailto:browne@cs.utexas.edu)

[27] J. L. Hammond, T. Minyard, and J. Browne, "End-to-end framework for fault management for open source clusters: Ranger," in Proceedings of ACM TeraGrid, no. 9, 2010.

[28] Scott, D. W. "Multivariate Density Estimation". Wiley, New York, 1992.